**Ngene Ugochukwu**

**Applied Project and Thesis Foundations**

1. Agriculture remains the backbone of many economies, yet underdeveloped countries continue to struggle with low crop yields due to outdated farming practices, poor resource management, and a lack of access to real-time agricultural insights. Many farmers rely on traditional "trial and error" methods, leading to inefficiencies in irrigation, fertilization, and crop rotation. The consequences include food shortages, financial instability for farmers, and vulnerability to climate change and natural disasters.

While developed nations benefit from advanced agricultural technologies and precision farming techniques, smallholder farmers in underdeveloped regions lack access to data-driven tools that could optimize production and sustainability. Existing solutions are either too expensive, require high technical expertise, or fail to provide localized recommendations based on soil quality, weather conditions, and crop performance.

1. **Functional Requirements:**

* User Management:
  + Farmers can register and log in to access their data.
  + Farmers can create profiles storing their name, address, geographical location, crops they work with, etc.
  + Admins can manage users and configure system settings.
* Picture uploading for analyses:
  + Farmers can quickly upload pictures of their crops to get AI-driven insights and recommendations on crop health.
* AI-Driven Recommendations:
  + The platform generates personalized irrigation, fertilization, and pest control plans.
  + AI suggests optimal crop rotation strategies based on historical data.
* APIs for External Integrations:
  + RESTful APIs built with C# (ASP.NET Core) allow third-party tools to access data.
  + Support for IoT integration (e.g., soil sensors for real-time monitoring).
* Real-Time Alerts & Notifications:
  + Farmers receive notifications for weather changes, pest threats, or irrigation needs.
  + The system can send SMS or email alerts for critical events.

1. **Non-Functional Requirements:**

* Scalability:
  + The system must efficiently handle large datasets and support a growing number of users (>1000).
  + Cloud-based architecture ensures adaptability to different regions.
* Performance:
  + The backend (C# & SQL) must provide millisecond response times for user queries and reports.
  + Real-time analytics should process data within seconds to ensure timely recommendations
* Security:
  + User data must be protected using encryption and secure authentication (JWT).
  + Role-based access control (RBAC) ensures only authorized users can modify data.
* Usability:
  + The UI must be intuitive, allowing farmers with minimal technical knowledge to use it.
  + Mobile-friendly design ensures accessibility from smartphones.

1. SRS and User Stories

**Scope**

CropWise provides the following functionalities:

* Crop Image Analysis: AI-based health assessment and recommendations.
* Real-Time Insights: Monitoring of soil moisture, weather conditions, and crop health.
* AI-Driven Crop Rotation Plans: Optimize planting cycles for soil sustainability.
* Pest & Disease Alerts: Early detection and prevention measures.
* Fertilization & Irrigation Recommendations: Personalized schedules for efficient resource use.
* Interactive Dashboard: Visualizing key agricultural metrics.
* Cloud-Based Storage: Secure and scalable storage of farm data.

User Stories/ use cases and their workflows

1. **Crop Image Analysis Workflow**

**Main Flow:**

1. The farmer logs into the CropWise platform.
2. Navigates to the "Crop Analysis" tab.
3. Uploads an image of a crop.
4. The system processes the image using an AI model.
5. AI returns analysis results with a diagnosis and recommendations.
6. The farmer reviews the analysis and applies corrective actions.

**Alternate Flow:**

* If the image quality is low, the system prompts the user to retake or upload a clearer image.
* If the AI model detects multiple possible conditions, it provides ranked recommendations for the farmer to choose from.

**Exception Flow:**

* If AI fails to process the image, the system displays an error message and suggests manual troubleshooting.
* If no recognizable crop is detected in the image, the system prompts the user to check the image and try again.

1. **AI-Driven Crop Rotation Planning Workflow**

**Main Flow:**

* 1. The farmer logs into the system.
  2. Selects "Crop Rotation Planning."
  3. Inputs data about current and past crops.
  4. AI generates an optimized rotation schedule.
  5. The farmer reviews and applies the suggested planting cycle.

**Alternate Flow:**

* If missing data is detected, the system suggests estimated values or prompts for manual input.
* If the user wants to override AI suggestions, they can input custom rotation preferences.

**Exception Flow:**

* If AI fails to generate a plan, the system provides a basic rotation guide based on pre-set rules.
* If internet connectivity is lost, the system saves inputs and retries processing once reconnected.

1. **Real-Time Alerts for Pests & Weather Workflow**

**Main Flow:**

1. The system continuously monitors weather data and pest activity.
2. If risk conditions are detected, an alert is triggered.
3. The farmer receives a notification via the dashboard and/or SMS.
4. Recommendations are provided on mitigation actions.

**Alternate Flow:**

* If the farmer dismisses an alert, the system logs it and allows for future review.
* Farmers can adjust alert sensitivity settings based on their preferences.

**Exception Flow:**

* If external data sources are unavailable, the system displays a "data unavailable" warning and suggests manual checks.
* If the notification system fails, alerts are queued and resent when the system is restored.

1. **Admin User Management Workflow**

**Main Flow:**

1. Admin logs into the CropWise platform.
2. Navigates to the **User Management** section.
3. Views a list of registered farmers and their access levels.
4. Can approve, suspend, or revoke user access as needed.
5. Admin confirms actions, and the system updates user status accordingly.

**Alternate Flow:**

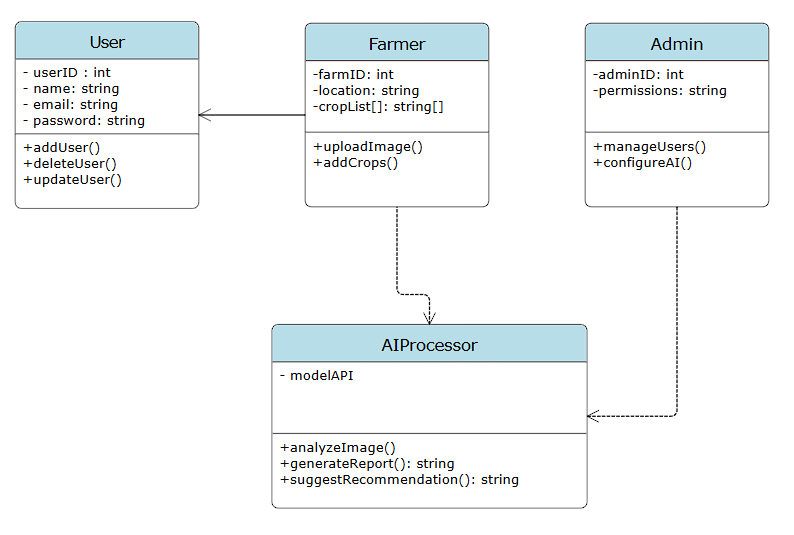
* If the admin wants to **add a new user manually**, they fill in user details and assign a role.
* If an inactive user requests reactivation, the admin reviews and restores access.

**Exception Flow:**

* If an admin tries to revoke a non-existent user, the system returns an error.
* If the system fails to update user status due to a database issue, an error message appears, and the admin is prompted to retry later.

**4.) Diagrams**

**Class Diagram :**



**Use case Diagram :**

